Polynomial network classifier with discriminative feature extraction

Cheng-Lin Liu

Polynomial Network Classifier with Discriminative Feature Extraction

Cheng-Lin Liu

National Laboratory of Pattern Recognition (NLPR)
Institute of Automation, Chinese Academy of Sciences
PO Box 2728, Beijing 100080, P.R. China
Email: liucl@nlpr.ia.ac.cn

Abstract. The polynomial neural network, or called polynomial network classifier (PNC), is a powerful nonlinear classifier that can separate classes of complicated distributions. A method that expands polynomial terms on principal subspace has yielded superior performance. In this paper, we aim to further improve the performance of the subspace-feature-based PNC. In the framework of discriminative feature extraction (DFE), we adjust the subspace parameters together with the network weights in supervised learning. Under the objective of minimum squared error, the parameters can be efficiently updated by stochastic gradient descent. In experiments on 13 datasets from the UCI Machine Learning Repository, we show that DFE can either improve the classification accuracy or reduce the network complexity. On seven datasets, the accuracy of PNC is competitive with support vector classifiers.

1 Introduction

Artificial neural networks (ANNs) with supervised learning have shown superior classification performance in many experiments [1]. Frequently used neural classifiers include the multi-layer Perceptron (MLP), radial basis function (RBF) network, polynomial network, etc. The polynomial network is also known as higher-order neural network (HONN), functional link network, polynomial regression [2], or generalized linear discriminant function [3]. In this paper, we call this classifier structure as polynomial network classifier (PNC). Since the outputs of PNC are the weighted combinations of higher-order nonlinear functions of input features, it is powerful to separate pattern classes of complicated distributions.

The PNC can be viewed as a single-layer neural network with the input features and their polynomial terms as the network inputs. For d features, the total number of polynomial terms up to r-th order is [4]

$$D = \sum_{i=0}^{r} \binom{d+i-1}{i} = \binom{d+r}{r}. \tag{1}$$

With large d, the polynomial network will suffer from high computation complexity and will give degraded generalization performance. The complexity can

be reduced by either reducing the number of input features or selecting expanded polynomial terms [2]. The former way is more computationally efficient and performs fairly well in practice. A PNC with dimensionality reduction by principal component analysis (PCA) has shown superior performance to multilayer neural networks in previous experiments [5, 6].

On the other hand, constrained polynomial structures with moderate complexity have been proposed, like the pi-sigma network (PSN) [7], the ridge polynomial network (RPN) [4], and the reduced multivariate polynomial model (RMPM) [8]. The general HONN is a sigma-pi network in that it combines the products of features. Rather, the output of a PSN is the product of weighted combinations of features. Its number of weights is thus linear with the number of summation units (the order of polynomials). The output of RPN is the summation of pi-sigma units of different orders, and the order can be increased incrementally. The RMPM combines the univariate polynomials, the polynomial of sum of features and its product with the weighted sum of features. These networks actually involve all the polynomial terms of input features up to certain order, but the weights of polynomials are highly constrained. They hence need polynomials of fairly high order (say, 5 or 6) to approximate complicated functions, and cannot guarantee the precision of approximation in difficult cases.

The PNC with full polynomials on reduced features still have higher complexity than the above constrained networks, but usually, a low order (say, 2 or 3) can achieve a reasonable precision of function approximation. The behavior of a lower-order network on feature subspace is easy to explain and to control. Nevertheless, its performance largely depends on the technique of feature selection or dimensionality reduction. Supervised subspace learning methods, like the Fisher linear discriminant analysis (LDA) [3] and heteroscedastic discriminant analysis [9, 10], may lead to better separability than the unsupervised PCA. These methods, nevertheless, are based on parametric density assumptions and the learning criterion is only loosely connected to classification error.

In this paper, we propose a subspace-feature-based PNC with discriminative feature extraction (DFE). With any classifier structure, DFE optimizes the subspace parameters together with the classifier parameters under a classification-related objective on a training sample set [11]. The subspace thus learned is totally classification-oriented and the subspace learning and classifier learning are best fitted. Overfitting can be overcome by adjusting the dimensionality of subspace and the order of classifier. DFE is mostly based on the minimum classification error (MCE) criterion of Juang and Katagiri [12], and has been successfully applied to many pattern recognition problems [13, 14]. It has not been combined with polynomial networks, however. Despite that the MCE criterion is applicable to any classifier structures, for neural networks with sigmoid outputs, the minimum squared error (MSE) criterion works well and is easy to optimize by stochastic gradient descent [15].

We have evaluated the classification performance of PNC on 13 datasets from the UCI Machine Learning Repository [16]. The results show that compared with the PNC with PCA, DFE either improves the classification accuracy or reduces

the network complexity. The complexity of PNC is much lower than support vector classifiers (SVCs) [17], and on seven of the 13 datasets, the PNC with DFE competes with SVCs in accuracy.

2 Subspace-Feature-Based PNC

We consider second-order (binomial) and third-order polynomial networks, and to save space, we only give the details of binomial networks. The structure and the learning algorithm of third-order networks are similar to binomial ones.

For M-class classification, the PNC has M output units. On a d-dimensional feature vector $\mathbf{x} = [x_1, \ldots, x_d]^T$, the output of binomial network for class ω_k is computed by

$$y_k(\mathbf{x}) = \sigma\Big[\sum_{i=1}^{d}\sum_{j=i}^{d} w_{kij}^{(2)} x_i x_j + \sum_{i=1}^{d} w_{ki}^{(1)} x_i + w_{k0}\Big], \tag{2}$$

where $\sigma(a)$ is the sigmoid activation function:

$$\sigma(a) = \frac{1}{1+e^{-a}}.$$

In classification, the input pattern (feature vector) is classified to the class of maximum output. The sigmoid function is used in training, and is not necessary in classification. Without the sigmoid function, the network weights can also be estimated by (non-iterative) pseudo inverse [2]. Since the sigmoid function makes the network outputs approximate posterior class probabilities, the trained weights with it are more suitable for classification than for regression.

By principal component analysis (PCA), the feature vector is projected onto an m-dimensional principal subspace ($m < d$):

$$\mathbf{z} = \Phi^T \mathbf{x} = [\phi_1^T \mathbf{x}, \ldots, \phi_m^T \mathbf{x}]^T = [z_1, \ldots, z_m]^T, \tag{3}$$

where $\Phi = [\phi_1, \ldots, \phi_m]$ is the transformation matrix (subspace basis) composed of the eigenvectors of covariance matrix $E[\mathbf{x}\mathbf{x}^T]$ corresponding to the m largest eigenvalues. We assume that the origin of the feature space has been shifted to the mean of samples. On the subspace features, the network outputs are computed by

$$y_k(\mathbf{x}) = \sigma\Big[\sum_{i=1}^{m}\sum_{j=i}^{m} w_{kij}^{(2)} z_i z_j + \sum_{i=1}^{m} w_{ki}^{(1)} z_i + w_{k0}\Big]. \tag{4}$$

On a training set of N samples $(\mathbf{x}^n, c^n)$, $n = 1, \ldots, N$ (c^n is the class label of $\mathbf{x}^n$), the connecting weights of PNC are adjusted to minimize the regularized squared error:

$$E = \frac{1}{2N}\Big\{\sum_{n=1}^{N}\sum_{k=1}^{M}\big[y_k(\mathbf{x}^n) - t_k^n\big]^2 + \beta \sum_{w\in W} w^2\Big\}, \tag{5}$$

where β is a coefficient of weight decay (excluding the biases); t_k^n is the target value of class k, with value 1 for the genuine class and 0 otherwise. The weights and biases are initialized to small random values, and by stochastic gradient descent, they are iteratively updated on the training samples until the squared error approaches the minimum. In training, the subspace basis Φ remains unchanged, and the polynomials of projected features can be viewed as the inputs of a single-layer network, for which the training process converges fast.

3 PNC with Discriminative Feature Extraction

A problem with the subspace-feature-based PNC is that the subspace does not necessarily lead to optimal classification because it is learned independently of the network weights. The subspace learned by PCA does not even consider the class information of training samples. Supervised subspace learning techniques, like LDA and heteroscedastic discriminant analysis, are expected to give better separability than PCA, but do not guarantee the optimality. We aim to learn a better subspace for PNC using discriminative feature extraction (DFE) [11].

By DFE, we adjust not only the network weights in supervised learning, but also the subspace basis simultaneously. Consider that $z_i = \phi_i^T \mathbf{x}$, $i = 1, \ldots, m$, let us re-write the network outputs of (4) as

$$y_k(\mathbf{x}) = \sigma\Big[\sum_{i=1}^{m}\sum_{j=i}^{m} w_{kij}^{(2)} \phi_i^T \mathbf{x} \phi_j^T \mathbf{x} + \sum_{i=1}^{m} w_{ki}^{(1)} \phi_i^T \mathbf{x} + w_{k0}\Big] = \sigma(s_k(\mathbf{x})), \qquad (6)$$

where $s_k(\mathbf{x})$ denotes the weighted sum of output unit k.

In the PNC with DFE, since the projected feature $z_i = \phi_i^T \mathbf{x} = \sum_{j=1}^{d} \phi_{ij} x_j$ is a weighted combination of original features and the weights (subspace parameters ϕ_{ij}, $j = 1, \ldots, d$) are now adjustable, an m-th order polynomial as $\prod_{i=1}^{m} (\phi_i^T \mathbf{x})$ is actually a pi-sigma unit of the ridge polynomial network (RPN). However, our network has more polynomial terms and needs a lower order than the RPN. Interpreting ϕ_i, $i = 1, \ldots, m$, as subspace basis vectors or feature extractors, a lower-order polynomial network on this feature subspace has decision boundaries of moderate complexity.

The network weights and the subspace basis parameters are adjusted to minimize the regularized square error (5) on a training sample set. The subspace parameters can be initialized to small random values as the network weights. Alternatively, the subspace learned by PCA or LDA is a good start of parameter search. The weights and subspace parameters are then adjusted by stochastic gradient descent on training samples. At time t, the parameters are adjusted on a training sample $\mathbf{x}$ by

$$\begin{aligned}
w_{kij}^{(2)}(t+1) &= w_{kij}^{(2)}(t) - \epsilon(t)\big[(y_k - t_k) y_k (1 - y_k) z_i z_j + \tfrac{\beta}{N} w_{kij}^{(2)}(t)\big], \\
w_{ki}^{(1)}(t+1) &= w_{ki}^{(1)}(t) - \epsilon(t)\big[(y_k - t_k) y_k (1 - y_k) z_i + \tfrac{\beta}{N} w_{ki}^{(1)}(t)\big], \\
w_{k0}(t+1) &= w_{k0}(t) - \epsilon(t)(y_k - t_k) y_k (1 - y_k), \\
\phi_i(t+1) &= \phi_i(t) - \epsilon(t) \textstyle\sum_{k=1}^{M} (y_k - t_k) y_k (1 - y_k) \frac{\partial s_k}{\partial \phi_i}, \\
& \qquad k = 1, \ldots, M,\ i = 1, \ldots, m,\ j = i, \ldots, m,
\end{aligned} \qquad (7)$$

where $\epsilon(t)$ is the learning step, which is set to a small value initially and decreases gradually in the training process. The partial derivative of ϕ_i is specified as

$$\frac{\partial s_k}{\partial \phi_i} = \Big(2w_{kii}^{(2)} z_i + \sum_{j<i} w_{kji}^{(2)} z_j + \sum_{j>i} w_{kij}^{(2)} z_j + w_{ki}^{(1)}\Big)\mathbf{x}. \tag{8}$$

In discriminative learning, we keep the unit norm of basis vectors but not the orthogonality. On adjusting the basis vectors on a training sample, each vector is normalized to unit norm ($\|\phi_i\| = 1$).

By stochastic gradient descent, the training samples are fed to the PNC for a number (40 or more in our experiments) of cycles. The learning step decreases linearly until it vanishes at the end of training. On every input sample, the network weights and subspace parameters are updated according to (7). The network weights and the subspace vectors have remarkably different magnitudes of derivatives. To accelerate the convergence of training, they are set two different learning steps, ϵ_1 for weights and ϵ_2 for subspace vectors, and $\epsilon_1 \gg \epsilon_2$ holds.

Another factor affecting training convergence and classification performance is the scale of projected features. We normalize the scale with the square root of the largest eigenvalue λ_1 of $E[\mathbf{x}\mathbf{x}^T]$ (estimated on training samples and fixed):

$$z_i = \frac{\phi_i^T \mathbf{x}}{\sqrt{\lambda_1}}. \tag{9}$$

All the feature vectors are subtracted from the mean of the training samples. For datasets that have significantly different scales among feature dimensions, it is helpful to uniform the standard deviation of all dimensions of training data (and test data accordingly). This is done before subspace projection.

4 Experiments

We evaluated the classification performance of subspace-feature-based PNC on 13 datasets from the UCI Machine Learning Repository [16], as summarized in Table 1. We selected the multi-class datasets that have at least 10 features. Some data sets have been partitioned into standard training and test subsets. For the others, we arrange the samples in random order and evaluate in 5-fold cross-validation.

Some datasets have appreciable variability of scale among different dimensions. We normalized them by dividing each dimension with $(0.9\sigma_i^2 + 0.1\sigma_0^2)^{1/2}$, where σ_i^2 is the dimension-wise variance and σ_0^2 is the average variance, both estimated on training data.

We compare the PNC-DFE (PNC combined with DFE) with PNC-PCA, one-versus-all support vector classifiers with polynomial and RBF kernels (SVC-poly and SVC-rbf), and the k-nearest neighbor (k-NN) classifier. For the SVC-poly, the feature vectors are uniformly scaled such that the average self-inner product of training vectors is one, and so, the kernel $k(\mathbf{x}_1, \mathbf{x}_2) = (1 + \kappa \mathbf{x}_1 \cdot \mathbf{x}_2)^r$ with $\kappa = 2^i$ performs fairly well. For the SVC-rbf, the average within-class variance

Table 1. Summary of 12 datasets from UCI Repository. The right two columns shows the selected polynomial order and subspace dimensionality (multiple of m_1).

Name	#class	#feature	#train	#test	Normal.	Order	m_1
Waveform	3	21	50,000	5-fold	No	2	1
Wine	3	13	178	5-fold	Yes	2	2
Soybean-small	4	35	47	5-fold	Yes	2	1
Vehicle	4	18	846	5-fold	Yes	2	2
Dermatology	6	34	358	5-fold	Yes	2	2
Segment	7	19	2,310	5-fold	Yes	3	3
Thyroid	3	21	3,772	3,428	Yes	2	4
Satimage	6	36	4,435	2,000	No	2	5
Optdigit	10	64	3,823	1,797	No	2	10
Pendigit	10	16	7,494	3,498	No	3	3
Vowel	11	10	528	462	No	2	2
Isolet	26	617	6,238	1,559	No	2	25
Letter	26	16	16,000	4,000	No	3	3

is scaled to one, such that in the kernel function $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\frac{\|\mathbf{x}_1-\mathbf{x}_2\|^2}{2\sigma^2})$, a parameter value of $\sigma^2 = 0.5 \times 2^i$ performs fairly well. For both, i is an integer selected from -4 to 4.

For the k-NN classifier, SVC-poly, and SVC-rbf, we tried several values of k, polynomial order and κ, or σ^2 such that the classification accuracy on each test set is maximized.

For PNC-PCA and PNC-DFE, we set the number of subspace features $m = l \cdot m_1$, $l = 1, \ldots, 5$. m_1 is dependent on the dataset. The selected values of polynomial order and m_1 are listed in the right columns of Table 1. As seen, three datasets (Segment, Pendigit and Letter) are used 2nd-order and the others are used 3rd-order.

The test accuracies (%) of PNC (with full polynomials and dimensionality reduction by PCA and DFE) on the 13 datasets are shown in Table 2. For the "Vowel" dataset, there is no dimensionality reduction when $m = 10$. For each dataset, the accuracy of full PNC is shown below the title of dataset, and the accuracies of PNC-PCA and PNC-DFE with variable subspace dimensionality are listed in two rows. For the "Isolet" dataset, we do not give the accuracy of full PNC because the number of features is too large.

We can see that on four datasets (Vehicle, Segment, Satimage, and Letter), the full PNC gives the highest accuracy. This can be explained that the four datasets have small number of features and are difficult to classify, so dimensionality reduction by either PCA or DFE cannot improve the classification accuracy. For the other datasets, except for "Soybean-small" and "Isolet", subspace-feature-based PNC performs significantly better than the full PNC.

Comparing the accuracies of PNC-PCA and PNC-DFE, it is evident that except for two datasets (Waveform and Satimage), PNC-DFE mostly give higher accuracies than PNC-PCA, especially on subspaces of lower dimensionality. On

Table 2. Test accuracies (%) of PNC (full and PCA) and PNC-DFE on 12 datasets.

Dataset	PCA=m_1	PCA=$2m_1$	PCA=$3m_1$	PCA=$4m_1$	PCA=$5m_1$
Full PNC	DFE=m_1	DFE=$2m_1$	DFE=$3m_1$	DFE=$4m_1$	DFE=$5m_1$
Waveform	63.78	87.02	**87.22**	87.12	86.98
84.92	60.64	86.90	86.96	86.72	86.64
Wine	77.53	90.45	92.13	92.70	92.70
92.13	79.79	92.13	**93.82**	93.82	93.82
Soybean-small	74.47	100	100	100	100
100	91.49	100	100	100	100
Vehicle	53.19	67.38	71.75	77.30	78.37
84.16	71.51	77.42	78.72	79.67	80.02
Dermatology	77.65	89.94	96.37	96.37	96.37
96.37	93.30	**96.93**	96.93	96.65	96.37
Segment	61.08	84.16	92.21	92.38	92.25
96.41	92.90	94.33	94.89	95.50	95.80
Thyroid	92.65	93.49	93.17	93.49	95.51
94.78	96.06	97.32	97.67	97.72	**97.87**
Satimage	86.75	87.10	87.75	88.00	87.95
88.65	86.45	87.45	87.90	88.15	88.05
Optdigit	95.72	98.05	98.61	98.61	98.55
98.50	97.16	98.50	**98.72**	98.50	98.66
Pendigit	85.11	95.77	97.68	98.03	**98.37**
98.23	89.57	97.17	97.91	98.17	**98.37**
Vowel	43.72	50.09	58.01	60.39	
59.52	57.14	60.17	**64.72**	61.47	
Isolet	93.33	95.19	95.51	**96.28**	96.28
	95.57	95.96	95.96	**96.28**	96.09
Letter	32.35	73.17	85.38	91.47	94.03
94.70	57.00	80.88	89.47	92.70	94.40

some datasets (Waveform, Soybean-small, Dermatology, Optdigit, Isolet), the PNC-DFE achieves the best or nearly best accuracy on a very low-dimensional subspace as $m = 2m_1$.

The highest accuracies of PNC (full and PNC-PCA), PNC-DFE, SVC-poly, SVC-rbf, and k-NN classifier on the 13 datasets are compared in Table 3. On the "Soybean-small" dataset, all these classifiers achieves perfect classification. Among the other datasets, SVC-poly or SVC-rbf gives the highest accuracies on seven datasets, and PNC or PNC-DFE performs best on five datasets. Expect for four datasets (Soybean-small, Segment, Satimage, and Letter), PNC or PNC performs significantly better than the k-NN classifier. The accuracy of PNC or PNC-DFE is comparable or higher than SVC on seven datasets (Waveform, Wine, Soybean-small, Vehicle, Thyroid, Optdigit, Vowel).

We did not implement the reduced multivariate polynomial model (RMPM) [8], but results on 10 of our 13 datasets were reported in the literature. Though

Table 3. Highest accuracies of PNC (full and PCA), PNC-DFE, SVCs and k-NN classifier.

	PNC	PNC-DFE	SVC-poly	SVC-rbf	k-NN
Waveform	**87.22**	86.96	87.14	87.08	85.24
Wine	92.70	**93.82**	92.13	93.26	87.08
Soybean-small	100	100	100	100	100
Vehicle	**84.16**	80.02	81.56	81.21	71.99
Dermatology	96.37	96.93	**97.77**	97.21	96.09
Segment	96.41	95.80	96.62	**96.88**	96.71
Thyroid	95.51	**97.87**	96.70	95.36	94.28
Satimage	88.65	88.15	90.70	**91.40**	90.35
Optdigit	98.61	98.72	98.66	**98.89**	98.00
Pendigit	98.37	98.37	**98.77**	98.74	97.80
Vowel	60.39	**64.72**	56.06	64.50	59.52
Isolet	96.28	96.28	**96.92**	96.86	92.69
Letter	94.70	94.40	96.78	**97.65**	95.83

the datasets were partitioned in different ways, nine of the 10 best accuracies of RMPM (Waveform 83.3%, Soybean-small 95.0%, Vehicle 82.3%, Segment 94.1%, Thyroid 94.0%, Satimage 88.2%, Optdigit 95.3%, Pendigit 95.7%, Letter 74.1%) are lower than our best accuracies of PNC or PNC-DFE.

The complexity of PNC mainly depends on the number of features, and is much lower than SVC and k-NN classifier. The k-NN classifier stores all training samples and compares them with each test pattern. The SVC has a large number of support vectors, ranging from 10% to 70% of all training samples. Due to the limited space, we do not discuss the computational complexity in details.

5 Conclusion

We proposed to improve the performance of subspace-feature-based polynomial network classifier (PNC) using discriminative feature extraction (DFE), which optimizes the subspace parameters together with the network weights on training samples. Under a regularized squared error criterion, the parameters are efficiently adjusted by stochastic gradient descent. In our experiments on 13 datasets of UCI Machine Learning Repository, DFE mostly improves the accuracy of subspace-feature-based PNC. At moderate complexity, the PNC (full or subspace-feature-based) outperforms the k-NN classifier on nine datasets and competes with support vector classifiers on seven datasets.

Acknowledgements

This work is supported by the Hundred Talents Program of Chinese Academy of Sciences. The author thanks the anonymous reviewers for valuable comments.

References

1. L. Holmström, P. Koistinen, J. Laaksonen, E. Oja, Neural and statistical classifiers—taxonomy and two case studies, *IEEE Trans. Neural Networks*, 8(1): 5-17, 1997.
2. J. Shürmann, *Pattern Classification: A Unified View of Statistical and Neural Approaches*, Wiley Interscience, 1996.
3. K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd edition, Academic Press, 1990.
4. Y. Shin, J. Ghosh, Ridge polynomial networks, *IEEE Trans. Neural Networks*, 6(3): 610-622, 1995.
5. U. Kreßel, J. Schürmann, Pattern classification techniques based on function approximation, *Handbook of Character Recognition and Document Image Analysis*, H. Bunke and P.S.P. Wang (Eds.), World Scientific, 1997, pp.49-78.
6. C.-L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Handwritten digit recognition: benchmarking of state-of-the-art techniques, *Pattern Recognition*, 36(10): 2271-2285, 2003.
7. Y. Shin, J. Ghosh, The Pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation, *Proc. 1991 IJCNN*, Seattle, Vol.1, pp.13-18.
8. K.-A. Toh, Q.-L. Tran, D. Srinivasan, Benchmarking a reduced multivariate polynomial pattern classifier, *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6): 740-755, 2004.
9. H. Brunzell, J. Eriksson, Feature reduction for classification of multidimensional data, *Pattern Recognition*, 33(10): 1741-1748, 2000.
10. M. Loog, R.P.W. Duin, Linear dimensionality reduction via a heteroscedastic extension of LDA: the Chernoff criterion, *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6): 732-739, 2004.
11. A. Biem, S. Katagiri, B.-H. Juang, Pattern recognition using discriminative feature extraction, *IEEE Trans. Signal Processing*, 45(2): 500-504, 1997.
12. B.-H. Juang, S. Katagiri, Discriminative learning for minimum error classification, *IEEE Trans. Signal Processing*, 40(12): 3043-3054, 1992.
13. X. Wang, K.K. Paliwal, Feature extraction and dimensionality reduction algorithms and their applications in vowel recognition, *Pattern Recognition*, 36(10): 2429-2439, 2003.
14. X. Yang, G. Pang, N. Yung, Discriminative training approaches to fabric defect classification based on wavelet transform, *Pattern Recognition*, 37(5): 889-899, 2004.
15. H. Robbins, S. Monro, A stochastic approximation method, *Ann. Math. Stat.*, 22: 400-407, 1951.
16. UCI Machine Learning Repository, http://www.ics.uci.edu/ mlearn/MLRepository.html.
17. C.J.C. Burges, A tutorial on support vector machines for pattern recognition, *Knowledge Discovery and Data Mining*, 2(2): 1-43, 1998.